Worksheet 3 Thinking procedurally Answers

1. You have been asked to write a procedure to count the number of vowels in a sentence. How can you ensure that the procedure will work for any length of sentence?

   By putting the characters of the sentence into an array or list, passing the array/list as a parameter to the subroutine and returning the number of vowels.

2. To implement a stack, you would need three procedures: InitialiseStack(stack), AddToStack(stack, item), RemoveFromStack(stack).

   The identifiers in brackets are **parameters** defined in the main program and passed to the procedure.

   What procedures and parameters would you need to be able to implement a queue?

   Students can probably work out that items are removed from the front of a queue and added to the rear. There may be a maximum size to the queue. They should be able to come up with something similar to:

   InitialiseQueue(queue, maxNumberInQueue), (set up empty queue with numberInQueue = 0 and front and rear pointers set to 0.)

   AddToQueue (queue, item)

   RemoveFromQueue(queue)

3. A hierarchy chart can be compared to an upside-down tree, with the root at the top and branches and leaves spreading downwards.

   The "leaves" are the lowest level modules and all or most of the detailed program code will be in the "leaves".

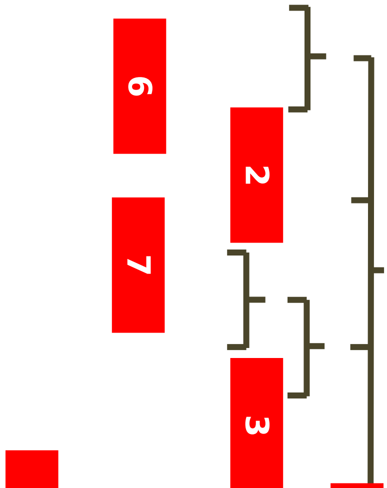
   In the hierarchy chart below:

   (a) Which are the Level 1 modules? 2, 3, 4, 5

   (b) Which are the Level 2 modules? 6, 7 ,8, 9

   (c) Which are the Level 3 modules? 10, 11

   (d) Write down the order in which the modules are executed. 6 7 3 10 11 9 5



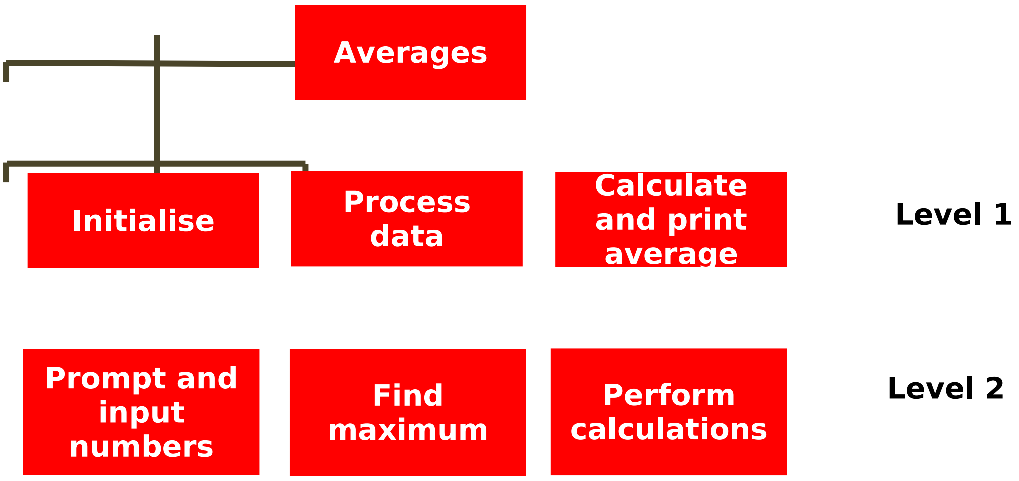
4. What are [redacted]e advantages of structured programming?
   - Each module can be individually tested
   - Modules ca[redacted]e re-used several times in a program
   - Frequently [redacted]d modules can be saved in a library and used by other programs
   - Several pro[redacted]mmers can simultaneously work on different modules, shortening development time
   - Structured programs are easier to understand and maintain

5. The following pseudocode program is designed to allow the user to input a series of three numbers and for each set of numbers, find and output the maximum. The maximum is then added to a total. When the user enters 0 for each of the three numbers, the average of all the maximums is calculated and output.

```
SUB initialise
    OUTPUT “This program finds the maximums of sets of three numbers.
            Enter three zeroes when all numbers entered.
            Program then calculates and outputs the average of the maximums”
    total = 0
    n = 0
ENDSUB


SUB promptForNumbers
    OUTPUT"Please enter first number "
    num1 = USERINPUT
    OUTPUT ("Please enter second number "
    num2 = USERINPUT
   OUTPUT "Please enter third number "
    num3 = USERINPUT
ENDSUB

SUB findMax
    maxnum = num1
    IF num2>maxnum THEN
        maxnum = num2
   ELSE
     IF num3>maxnum THEN
        maxnum = num3
    ENDIF
    OUTPUT "Max of the three numbers is is ",maxnum
ENDSUB

SUB performCalculations
   total = total + maxnum
   n=n+1
ENDSUB

SUB processData
    promptForNumbers
    WHILE num1<>0 and num2<>0 and num3<>0
        findMax
        performCalculations
        promptForNumbers
    ENDWHILE
ENDSUB

SUB calculateAverage
    average = total/n
    OUTPUT "Average of maximums is ",average
```

```
ENDSUB

#Main program starts here
initialise
processData
calculateAverage
```

Draw a hierarchy chart representing this program. Show the different levels, i.e. Level 1 modules, Level 2 modules etc.



see Python programs Worksheet 3 maximums.py and Worksheet 3 maximums.vb

worksheet 3 max as subs.py and worksheet 3 max as subs.py